A Framework for the Automation of Air Defence Systems*

Sunil Choenni Kees Leijnse

National Aerospace Lab., NLR
P.O. Box 90502
1006 BM Amsterdam
The Netherlands
email: {choenni, leijnse}@nlr.nl

Abstract

The need for more efficiency in military organizations is growing. It is expected that a significant increase in efficiency can be obtained by an integration of communication and information technology. This integration may result in (sub)systems that are fully automated, i.e., systems that are unmanned, including unmanned vehicles. In this paper, we focus on the automation of air defence systems, in which integration of communication and information technology is a major issue. We propose an architecture, in which each weapon system has the capability to control itself, whilst acting in a co-ordinated manner with other systems. To realise this task, a weapon system is exactly informed about the activities of all other weapon systems. In our architecture, the role of the men is reduced to the supervision of weapon systems.

1 Introduction

While communication technology is an integral part of military systems, the potentials of information technology have recently been recognized by military organizations. Since it has been demonstrated that information technology provides the possibility to facilitate or to automate a wide variety of tasks that are currently performed by military experts, military organizations are rapidly adopting this technology. Exploiting information technology may lead to a decrease in the number of military personnel required for such tasks, and to an increase of the efficiency in military organizations. In [5, 6], the need for more efficiency in military organizations has been discussed. It is expected that a further increase in efficiency can be obtained by the integration of communication and information technology. This integration may result in (sub)systems that are fully automated, i.e., systems that are unmanned, including unmanned vehicles.

We are interested in the application of information and communication technology and in the impact of these technologies on air defence systems. An air defence system has as goal the defence of a predefined space against physical attack and espionage from the air. To realise this task, air defence systems are equipped with a wide variety of means, such as men, weapons, vehicles, sensors, etc. On the basis of the size of the space that should be defended, NATO distinguishes four categories of air defence systems namely, very short range air defence systems (vshorad), short range air defence systems (shorad), medium range air defence systems, and air defence fighters. Very short range air defence systems defend spaces that range up to 6 kilometres in a horizontal direction and up to 3 kilometres in a vertical direction. For short range air defence systems these sizes are 12 and 6 kilometres in horizontal and vertical direction respectively. For longer distances the remaining categories are used.

In this paper, we focus on the automation of vshorad and shorad systems. Apart from military experts, these systems basically consist of a set of sensors and a set of kill vehicles, which may be located on geographically different bases. Sensors are used to detect incoming targets and to track these targets. Kill vehicles are assigned to destroy targets. A combination of sensors and kill vehicles is called a weapon system, and a shorad/vshorad system can be regarded as a set of weapon systems that are controlled by military experts. Each weapon system is dedicated to the defence of a part of the space assigned to a vshorad/shorad system.

We propose an architecture, referred to as *distributed* architecture, in which each weapon system has the capability to control itself in a co-ordinated manner. This architecture is based on two principles. First, each weapon system has access to the same set of data and the same capabilities to process this data. Second, each weapon[1] and sensor knows the strategies that are used to deploy sensors to observe an area, i.e., a part of the airspace, and to allocate weapons to targets. For example, strategies for sensor deployment and weapon allocation may be that an area is

*This research has been performed within the scope of the NATO SHORAD/VSHORAD Feasibility Study in the Matra BAe Dynamics consortium.

[1] In the following, the terms weapon and kill vehicle are used interchangeably.

Paper presented at the RTO SCI Symposium on "Warfare Automation: Procedures and Techniques for Unmanned Vehicles", held in Ankara, Turkey, 26-28 April 1999 and published in RTO MP-44.

observed by the closest located sensor and a target is attacked by the closest located weapons, respectively. These principles have as consequence that each sensor or weapon may know exactly what all other sensors and weapons are doing in the system, and can act in a co-ordinated manner. In our architecture, the role of the men is reduced to the supervision and maintenance of the system.

The distributed architecture is only viable if the technology to handle the two principles is sufficiently mature, and we believe that this is the case. To handle the first aspect of the first principle, that is, to provide each weapon system the access to the same set of data, we propose a network to which all entities, including other sensors and weapons, are connected. Each sensor/weapon or other connected entity is able to extract data from the net, and is able to request net capacity (bandwidth/time slots) in order to put data on the net. The acceptance of the request and the capacity that will be allocated to an entity depends on the load of the network and on the importance of the data for other entities. So, dynamic allocation of net capacity is the proposed solution.

To handle the second aspect of the first principle, that is, to provide processing capabilities to each weapon system, the architecture should be equipped with algorithms to perform the tasks that are required for (very) short range air defence, such as data fusion, threat evaluation, weapon allocation, etc. In the literature, a wide variety of potential algorithms has been reported to perform these tasks, see [2, 3, 6, 9]. We propose a framework in which many of these algorithms can be captured. In this framework, we distinguish a pool of algorithmic skeletons and a pool of logical operators. An algorithmic skeleton consists of important control statements. By combining operators and algorithmic skeletons, an algorithm can be generated for a specific task.

It is clear that the second principle, i.e., each weapon system knows the strategies for sensor deployment and weapon allocation, can be handled with above-mentioned techniques as well.

The main advantages of our architecture are performance and reliability. Performance is achieved by the fact that a weapon system has its own processing capabilities and the possibility to load and organize data in an efficient way. Note, that a bad organization of data may lead to a poor performance of an overall system [1]. Reliability is achieved by the fact that each weapon system is informed about each other's activities, which avoids situations that a target is overkilled, or, even worse, that a target is not attacked at all. Other nice properties of the architecture are that it supports modularity and graceful degradation. We note that these latter two properties are also inherent to an autonomous architecture. The main difference between our architecture and systems based on an autonomous architecture, such as the US FAAD system, is that in the latter architecture weapon systems are not informed about each other's activities.

The remainder of this paper is organized as follows: in Section 2, we discuss the distributed architecture in more detail. Since communication and processing algorithms play a major role in this architecture, the two consecutive sections 3 and 4 are devoted to them. Finally, the paper is concluded in Section 5.

2 Distributed Architecture

Our framework to automate air defence systems is based on the concept that each weapon system has the capability to control itself in a coordinated manner. Therefore, we propose an architecture in which all entities are connected to a network. As soon as an entity obtains new information/data, it puts it on the network. All other entities have the possibility to access this information/data. In order to realize that entities act in a coordinated manner, all entities have the same processing algorithms. So, processing of the same data will result in the same results at each entity, given instantaneous differences due to time delays.

The entities that are distinguished for the time-being are weapon systems and command centres. The basic architecture is depicted in Figure 1.

A weapon system consists of a set of kill vehicles and a set of sensors. Communication between kill vehicles and/or sensors is realised through the network. Typical information that will be put on the net by kill vehicles are plans to attack a target. Sensors will put measurements performed in the real world on the network.

A command centre is hierarchically organised, consisting of three levels. A battalion at the highest level controls a set of batteries, and a battery in its turn controls a set of platoons. Each level is connected to the network. So, information from the battalion destinated for a battery can also be obtained by a platoon. Although each entity in a specific level has all available information and processing capabilities to take justified decisions, the reason to preserve the hierarchical organization in command centres is that a higher level echelon should have the possibility to overrule a decision at a lower level echelon.

The different levels in a command centre are distinguished by the functions that are performed at each level. While the tasks to be performed at higher levels are strategical in character, at lower levels the tasks are more tactical. For example, a battalion is also connected with external systems and it may receive recognised air picture data from these systems. It is the responsibility of the battalion to select and distribute proper data to all entities through the network. At platoon level, weapons are commanded to attack a target.

The major advantages of distributing data to all entities through a network are reliability and performance. Reliability is achieved by the fact that each entity is informed about the activities of all other en-

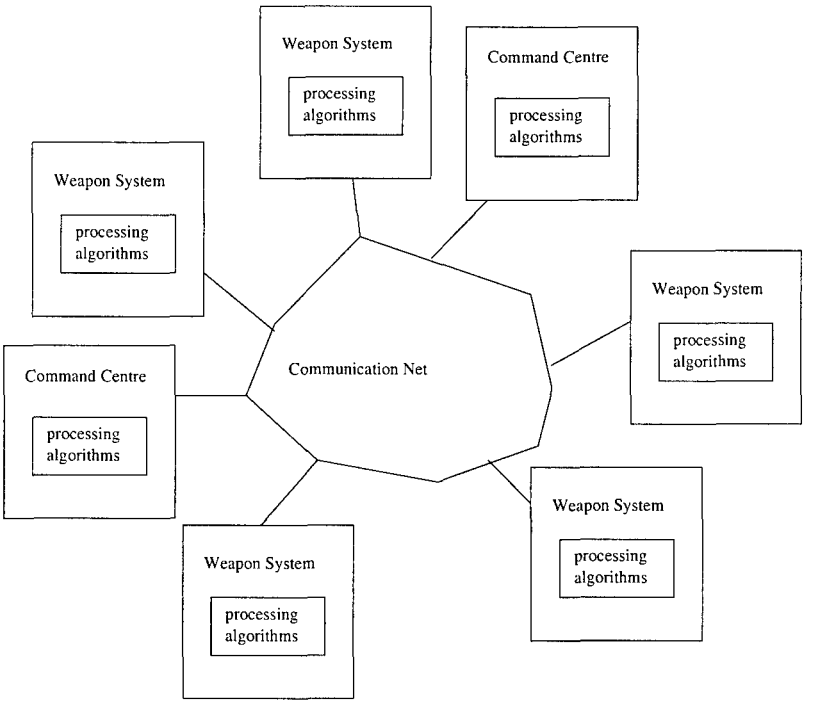

Figure 1: Basic architecture

tities and that each entity is capable to obtain and process data. By informing entities about their activities, situations of over-kill or not engaging a target can be avoided. For example, suppose that w_1 is the most obvious weapon to engage a target, but is unable to fulfil this task for some reason. Since the second obvious weapon, let's say w_2, can observe that w_1 is not going to engage the target, w_2 knows that it should engage the target.

By providing each entity processing capability, entities become independent of each other. So, they do not suffer from entities that fail to perform processing tasks. Let us consider the following situation for track correlation. Suppose that only one entity is able to perform track correlation and is also responsible for putting updated tracks on the network. If this entity fails to perform this task, then the entities have an obsolete track. Furthermore, the measurements provided by sensors in this case can be considered as a waste of effort. Another advantage of providing processing capability to each entity is that even if an entity fails to process some data, it still can obtain the results of processing, since other entities have processed the data, and may put it on the net.

Performance is achieved by the fact that there is a minimum delay in obtaining data, since all data is freely available. Since each entity has its own processing facilities, the queues for processing an amount of data will be much shorter compared to the situation, in which there would be one processing unit and each entity was assigned to this unit. Furthermore, processing algorithms may be tuned towards the tasks that should be performed by an entity, e.g., by incorporating specific domain knowledge in the algorithms. An additional advantage of providing each entity processing facilities is that graceful degradation is supported. This means that if some entities are completely destroyed, the other entities can still perform their tasks.

Since an enormous amount of data may flow through the network, congestion of the network is an obvious possibility in this architecture. In the next section, we describe a method to prevent and to cope with congestion.

3 Communication

In the proposed architecture (Figure 1), relevant data need to be shared between entities in an "all know everything" setup. Therefore data generated by one entity (e.g. relating to the detection of air targets by a sensor) should be available nearly instantaneously throughout the system, e.g., for track correlation, multi-sensor data fusion, threat evaluation, etc. This obviously calls for high-capacity data transmission between the system's elements. However, in multi-element wireless communication, capacity usually is limited.

A military network's data throughput capacity is embodied in time slots and frequencies/bandwidths embedded in a cyclic framework. A well-known representation (derived from the system Link-16 protocols) is shown in Figure 2. At the setup of such a (secure) communications network, each element is allotted an appropriate number of slots within the cycle in which it may transmit, at the prescribed hopping frequencies. In practice, this method leads to non-optimal usage of network capacity, as slots are allotted to sys-

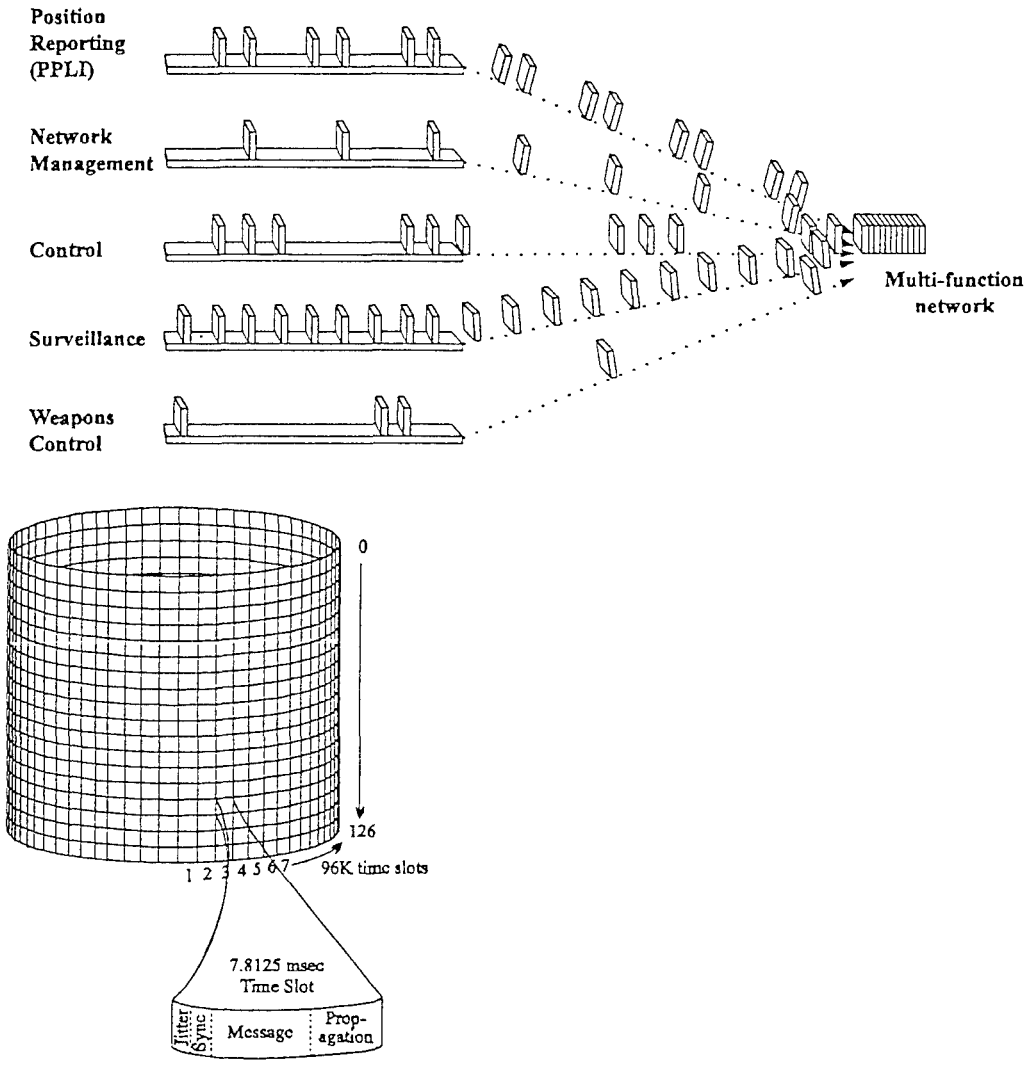

Figure 2: Cyclic representation of frequency bands vs. time slots

tem elements regardless of the volume of data generated and also regardless of the relevance of such data to the task. E.g., sensors having nothing of interest to report still consume part of the communications capacity.

In a typical air assault scenario, some entities (e.g. the upfront sensors) may generate lots of relevant data as they "see" many new targets. Such information may consist of track data as well as of signature characteristics. If the number of transmission slots originally allotted to those sensors is insufficient for transmission of all of the relevant information, optimal application of network capacity requires allocation of extra resources for the timely dispersion of such data.

A method of optimizing transmission capacity in a distributed network is proposed here. It is called the dynamic allocation of slots, frequencies and bandwidths. In this method, each system element requiring to transmit extra data first submits a transmission request comprising a weighted assessment of the urgency of its data. Since in the proposed architecture knowledge is dispersed throughout the system and entities possess the intelligence to decide whether the information they generate merits putting it on the net, the assessment of urgency can be made. Typical parameters that play a role in such an assessment are new target recognition results, optimum kill probability, time left to last launch opportunity, value of the threatened asset, etcetera. The extra slots (frequencies, bandwidths) are allocated dynamically, through a distributed management function and in proportion to the "weights" of the current requests. This approach is thought to be feasible, since successful time sharing schemes for mainframe computer operating systems are based on a similar concept. We further believe the method is promising, as it may generate a more efficient application of a scarce commodity.

We note that the proposed method should be complementary applied to (existing) data compressing techniques. It has been proven that data compressing techniques may considerably reduce a piece of data that should be transmitted. Once the transmitted data has been received, it may be decompressed such that the semantic of the original data is preserved.

4 Processing algorithms

In Figure 1, it is depicted that processing algorithms are required by weapon systems as well as by command centres. Within a command centre algorithms are required at all levels for various tasks. Typical tasks that may be performed by algorithms are track filtering and correlation (in order to produce air pictures), identification, threat evaluation, weapon allocation, etc. While some of these tasks should be performed by most entities and at each level, other tasks are performed just at some levels within a command centre. For example, an air picture is built up by all entities and on each level. Therefore, all entities and all levels should be equipped with track filtering, correlation, and data fusion algorithms. Intelligence gathering is a task that will typically be performed at battery and/or battalion level. So algorithms that support this task should be installed at these levels.

Many algorithms reported in the literature can be used to perform a wide range of tasks required for air defence systems. As an example, let us consider the task of track filtering and potentially useful algorithms for this task that can be found in text books. The core of track filtering is deciding whether a point is within or outside a polygon. Track filtering provides the possibility to remove air tracks that are outside a geographical area of interest.

Possible solutions for track filtering might be based on, e.g., point inclusive algorithms or nearest neighbour algorithms. The idea behind point inclusive algorithms is to draw a vertical or horizontal line from a point to the polygon, while counting the number of intersections between the line and the polygon. An even number of intersections implies that the point is outside the polygon, while an odd number implies that the point is inside the polygon. Intersections that are also a point of contact are counted as two intersections. The idea behind nearest neighbour algorithms is to assign a point to the most likely polygon. Therefore, these algorithms compute for all (relevant) polygons the probability that a point is within a polygon.

What algorithm to select for a task depends on the characteristics of the task and the available input. In general, each algorithm will have its own strong and weak points. For example, a point inclusive algorithm may be very fast, but on the other hand, it requires a detailed geometrical description of the polygon.

In the next section, we propose a framework that captures a wide variety of algorithms that may be used for several tasks by air defence systems.

4.1 Framework

Our main goal is to develop and implement algorithms that may be used for several air defence tasks. It has been widely recognised that the development of software for complex systems is a tough process. Therefore, several methodologies have been developed to facilitate this task at various levels, ranging from the design to the implementation. For example, data-driven, object-oriented, and top-down functional methodologies are well known at the design level, and for programming purposes the top-down and bottom-up methodologies are well known. Depending on the nature of an application, software engineers choose a number of these methodologies to develop software. For the software development of the proposed air defence system architecture, we will not design software from scratch but attempt to tailor existing algorithms for various functions. In general, the pseudo-code of these algorithms can be found in textbooks. Tailoring an algorithm to a function boils down to, e.g.,

- Verifying whether the assumptions on which an algorithm is based are realistic for the function or not, e.g., is the input expected by the algorithm available?
- What is the best way to represent the input for the algorithm? The representation should be such that it fits the problem domain, i.e., the problem that should be solved by the function.
- Are all operators in the algorithm meaningful? If not, should they be modified, or deleted?
- Are the control statements in the algorithm meaningful or should they be modified, or deleted?
- How should the output be represented?

We note that performing the above-mentioned tasks successfully requires advanced skills of a software engineer.

On the one hand, we have observed that several (textbook) algorithms might be used for a specific air defence task, while on the other hand an algorithm might be used for several air defence tasks. For example, a point inclusive algorithm as well as a nearest neighbour algorithm can be used for track filtering, while the latter algorithm can also be used for identification and threat evaluation. Our goal is to come up with a set of algorithms such that a single algorithm might be used for several air defence systems tasks on the one hand and on the other hand, we prefer to have several algorithms available to perform a task. To realise this goal, we suggest to implement a set of algorithmic skeletons and a set of operators. In an algorithmic skeleton, important control statements are implemented and operators are implemented in an abstract way. In a separate pool, operators are implemented in more detail. An operator describes how objects should be represented and what its impact will be on each object. Once these two sets are available, a user may construct its own algorithms by combining skeletons and operators. In this way, we re-use software as much as possible.

We note that for many textbook algorithms the distinction between algorithmic skeletons and operators can easily be made. Observe that an algorithm can be regarded as an ordered list of control statements and operations.

Once an algorithm has been constructed by combining operators with an algorithmic skeleton, this algorithm has to be instantiated. This means that values for the input parameters should be made available to the algorithm. Then, the algorithm can be compiled and executed. In Figure 3, the whole process is depicted.

The main advantage of our framework is that there are several alternative algorithms to perform a task, each with its own strong and weak points. If the result of an algorithm is unsatisfactory, one may assemble another algorithm.

4.2 An example

In this section, we illustrate our framework by means of a simplified identification algorithm. Identification algorithms collect data/evidences from multiple sources and combine these data in order to produce a composite identification of an object. Potential sources of data include recognised air pictures, procedural indicators (e.g., restricted area violations), acoustic sources, etc.

In our example, the goal is to determine what objects are in the airspace on the basis of a sequence of independent evidences. To solve this problem, we will discuss two techniques that might be used namely, one emanated from probability theory [7] and the other emanated from Dempster-Shafer theory [8]. Both theories provide us a tool to combine several bodies of evidence. For the similarities and differences between these theories, we refer to [4]. In the following, we will stress the combination of evidences.

In the airspace, we want to distinguish between civil aircraft, military aircraft, and birds. The set $D = \{civil\ aircraft,\ military\ aircraft,\ bird\}$ is called the frame of discernment. As time went on, evidences will be collected that support or reject a subset of D.

Let $D' \subseteq D$, and $P(D'|e_n)$ be the probability in D' given a sequence of $e_1, e_2, e_3, ..., e_n$ evidences. To update the probability in D', whenever a new body of evidence e becomes available, the following formulae can be used according to probability theory.

$$P(D'|e_n, e) = P(D'|e_n)\frac{P(e|D')}{P(e)}$$

and

$$P(D'|e_1) = \frac{P(e_1|D')P(D')}{P(e_1)}$$

Note that in the formula above we assumed that evidences are independent of each other.

Before introducing the rule to combine evidences according to the Dempster-Shafer theory, we introduce the notion of basic probability assignment. A basic probability assignment to a set D', $m(D')$ can be regarded as the measure of belief that is exactly committed to D'. A basic probability assignment should satisfy the following properties $m(\emptyset) = 0$ and $\sum_{D' \subseteq D} m(D') = 1$.

Let $m_{e_n}(.)$ be the basic probability assignment induced by a sequence of evidences $e_1, e_2, e_3, ..., e_n$. To update the belief in a set D', whenever a new body of evidence e becomes available the following formula can be used.

$$m_{e_n} \oplus m_e(D') = K^{-1} \sum_{\substack{i,j \\ D_i \cap D_j = D'}} m_{e_n}(D_i) m_e(D_j)$$

in which D' is a non empty set, $D_i, D_j \subseteq D$, and

$$K = \sum_{\substack{i,j \\ D_i \cap D_j \neq \emptyset}} m_{e_n}(D_i) m_e(D_j)$$

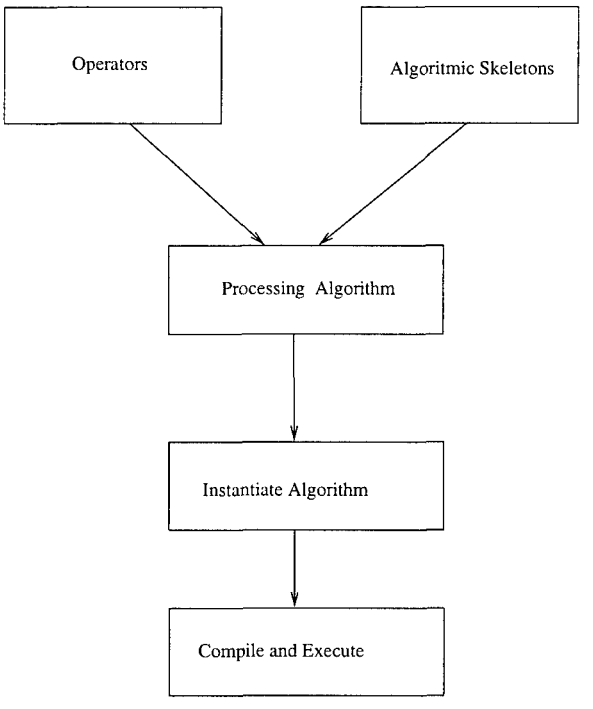

Figure 3: Framework to capture algorithms

We note that K is a normalization constant and is required to meet the property $m(\emptyset) = 0$.

Each of the above mentioned techniques can be implemented as a separate combination operator. Let our pool of operators consists of Combine_Prob, which is based on probability theory and Combine_DS, which is based on Dempster-Shafer theory.

Assume that the following algorithmic skeleton is available, in which a Combine operator appears.

```
Program Skeleton(D, Var Concl);
Get(e); /* Get reads a body of evidence /*
first_evidence := 'true';
while e ≠ '' do
      j:= 0;
      while j < 2^D do
            Combine(D'[j], first_evidence, e, Concl);
                /* Combine updates the belief/
                   likelihood in D'[j] /*
            j:= j+1;
      od;
      Get(e);
      if first_evidence = 'true'
      then first_evidence := 'false';
od;
end.
```

This skeleton requires as input a frame of discernment D, e.g., $D = \{civil\ aircraft,\ military\ aircraft,\ bird\}$. The output, referred to as *Concl*usion, will be subset(s) of D to which a measure is attached expressing the belief/likelihood that an object can be identified which one of the elements in the subset(s). As long as evidences are available, the belief/likelihood in each subset $D' \subseteq D$ is updated by the skeleton. Suppose that the a priori probabilities $P(e)$ and the a posteriori probabilities $P(e|D')$, in which $D' \in \{civil\ aircraft,\ military\ aircraft,\ bird\}$ and the goal is to identify whether an object is a bird, a civil or a military aircraft. Then, the above mentioned skeleton together with the Combine_DS operator can be used for this purpose. Now the identification program will be

```
Program Identification(D, Var Concl);
Get(e);
first_evidence := 'true';
while e ≠ '' do
    j:= 0;
    while j < 2^D do
      Combine_Prob(D'[j], first_evidence, e, Concl);
      j:= j+1;
    od;
    Get(e);
    if first_evidence = 'true'
    then first_evidence := 'false';
od;
end.
```

Once we have specified the input values, i.e., D, the required probabilities, the program is instantiated and ready for execution.

We note that if both probabilities $P(e)$ and $P(e|D')$ are not available, we have the possibility to build in the Combine_DS into the skeleton, resulting in an alternative identification program.

Suppose that our pool of operators contains a combination operator that is able to combine images, i.e., a body of evidence results in an image of an environment and we are able to combine different images, the same skeleton may be used for threat evaluation.

Summarising, we propose a framework that consists of a pool of operators and a pool of algorithmic skeletons. An operator manipulates a number of objects according to a certain technique. An algorithmic skeleton consists of control statements and abstractly defined operators. Now, an algorithm may be constructed by combining skeletons with operators. In this way, operators and skeletons can be used for several air defence tasks, and several alternatives will be available for a single air defence task.

5 Conclusions & further research

We have discussed a framework for the the automation of air defence systems. In this framework, the integration of information and communication technology is a major issue. We have proposed a distributed architecture, in which each weapon system has the capability to control itself in a co-ordinated manner. In this architecture, a weapon system is exactly informed about the activities of all other weapon systems. We have touched on how our architecture can be implemented using information and communication technology. Communication between entities and adequate processing of data by each entity are of vital importance. Communication between entitities is realised through a communication net, and net capacity is dynamically allocated to entities. For the processing of data, we have proposed to implement a wide variety of algorithmic skeletons and operators. An algorithm to perform an air defence task may be constructed by combining algorithmic skeletons and operators.

Furthermore, we have discussed the advantages of our architecture in relation with other architectures.

Topics for further research are the implementation and evaluation of the architecture.

Acknowledgement Dr. Hein Veenhof of NLR is thanked for his valuable comments on earlier drafts of this paper.

References

[1] Choenni, R., H.M. Blanken, H. Wagterveld, Automating Physical Database Design, Handbook of Data Management 1998, Thuraisingham, B. (Ed.), Auerbach Publications, Fl, USA, 1998.

[2] Choenni, R., On Vehicle Allocation to Targets in Mission Planning, RTA SCI 9th Symp. on the Application of Information Technologies to Mission Systems, NATO, RTA Publications MP-3, 1998.

[3] Donker, J., Reasoning with Uncertain and Incomplete Information in Aerospace Applications, AGARD Proc. on Machine Intelligence for Aerospace Electronic Systems, NATO, AGARD CP499, 1991.

[4] Halpern, J.Y., Fagin, R., Two Views of Belief: Belief as Generalized Probability and Belief as Evidence, Artificial Intelligence 54(3), 1992.

[5] Klomp, J., Zetten, H. van, Army Organic Air Defense: Effective and Affordable after 2000?, Militaire Spectator 167(3), 1998 (in Dutch).

[6] Krogmann, U., Towards Autonomous Systems, in AGARD Lecture Series 210, Advances in Soft-Computing Technologies and Application in Mission Systems, NATO, AGARD LS 210, France, 1997.

[7] Pearl, J., Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers, San Mateo, CA, USA, 1988.

[8] Shafer, G., A Mathematical Theory of Evidence, Princeton University Press, Princeton, USA, 1976.

[9] Wal, A. van der, The potential of soft-computing methods for mission systems, RTA SCI 9th Symp. on the Application of Information Technologies to Mission Systems, NATO, RTA Publications MP-3, 1998.